Python - 100 Numpy Exercises "A joint effort of the numpy community"

Python is one of the best tools for Data Science. And, Numpy is one of the best Python libraries for data processing.

This is a collection of exercises that have been collected from NumPy documentation and stackoverflow. This document covers 100 Numpy exercises and their solutions.

It should be very useful for learning data processing in Python as well as for interviews.

Note: If you find an error or think you've a better way to solve some of them, feel free to open an issue at https://github.com/rougier/numpy-100

Keep learning. Keep sharing. Keep growing. All the best!

1. Import the numpy package under the name `np` (★☆☆)

```
import numpy as np
```

2. Print the numpy version and the configuration (★☆☆)

```
print(np.__version__)
np.show_config()
```

3. Create a null vector of size 10 (★☆☆)

```
Z = np.zeros(10)
print(Z)
```

4. How to find the memory size of any array (★☆☆)

```
Z = np.zeros((10,10))
print("%d bytes" % (Z.size * Z.itemsize))
```

5. How to get the documentation of the numpy add function from the command line? (★☆☆)

```
%run `python -c "import numpy; numpy.info(numpy.add)"`
```

6. Create a null vector of size 10 but the fifth value which is 1 (★☆☆)

```
Z = np.zeros(10)
Z[4] = 1
print(Z)
```

7. Create a vector with values ranging from 10 to 49 (★☆☆)

```
Z = np.arange(10,50)
print(Z)
```

8. Reverse a vector (first element becomes last) (★☆☆)

```
Z = np.arange(50)
Z = Z[::-1]
print(Z)
```

9. Create a 3x3 matrix with values ranging from 0 to 8 (★☆☆)

```
Z = np.arange(9).reshape(3, 3)
print(Z)
```

10. Find indices of non-zero elements from [1,2,0,0,4,0] (★☆☆)

```
nz = np.nonzero([1,2,0,0,4,0])
print(nz)
```

11. Create a 3x3 identity matrix (★☆☆)

```
Z = np.eye(3)
print(Z)
```

12. Create a 3x3x3 array with random values (★☆☆)

```
Z = np.random.random((3,3,3))
print(Z)
```

13. Create a 10x10 array with random values and find the minimum and maximum values (★☆☆)

```
Z = np.random.random((10,10))
Zmin, Zmax = Z.min(), Z.max()
print(Zmin, Zmax)
```

14. Create a random vector of size 30 and find the mean value (★☆☆)

```
Z = np.random.random(30)
m = Z.mean()
print(m)
```

15. Create a 2d array with 1 on the border and 0 inside (★☆☆)

```
Z = np.ones((10,10))
Z[1:-1,1:-1] = 0
print(Z)
```

16. How to add a border (filled with 0's) around an existing array? (★☆☆)

```
Z = np.ones((5,5))
Z = np.pad(Z, pad_width=1, mode='constant', constant_values=0)
print(Z)

# Using fancy indexing
Z[:, [0, -1]] = 0
Z[[0, -1], :] = 0
print(Z)
```

17. What is the result of the following expression? (★☆☆)

```
0 * np.nan
np.nan == np.nan
np.inf > np.nan
np.nan - np.nan
np.nan in set([np.nan])
0.3 == 3 * 0.1
```

```
print(0 * np.nan)
print(np.nan == np.nan)
print(np.inf > np.nan)
print(np.nan - np.nan)
print(np.nan in set([np.nan]))
print(0.3 == 3 * 0.1)
```

18. Create a 5x5 matrix with values 1,2,3,4 just below the diagonal (★☆☆)

```
Z = np.diag(1+np.arange(4),k=-1)
print(Z)
```

19. Create a 8x8 matrix and fill it with a checkerboard pattern (★☆☆)

```
Z = np.zeros((8,8),dtype=int)
Z[1::2,::2] = 1
Z[::2,1::2] = 1
print(Z)
```

20. Consider a (6,7,8) shape array, what is the index (x,y,z) of the 100th element? (★☆☆)

```
print(np.unravel_index(99,(6,7,8)))
```

21. Create a checkerboard 8x8 matrix using the tile function (★☆☆)

```
Z = np.tile( np.array([[0,1],[1,0]]), (4,4))
print(Z)
```

22. Normalize a 5x5 random matrix (★☆☆)

```
Z = np.random.random((5,5))
Z = (Z - np.mean (Z)) / (np.std (Z))
print(Z)
```

23. Create a custom dtype that describes a color as four unsigned bytes (RGBA) (★☆☆)

```
color = np.dtype([("r", np.ubyte),
                  ("g", np.ubyte),
                  ("b", np.ubyte),
                  ("a", np.ubyte)])
```

24. Multiply a 5x3 matrix by a 3x2 matrix (real matrix product) (★☆☆)

```
Z = np.dot(np.ones((5,3)), np.ones((3,2)))
print(Z)

# Alternative solution, in Python 3.5 and above
Z = np.ones((5,3)) @ np.ones((3,2))
print(Z)
```

25. Given a 1D array, negate all elements which are between 3 and 8, in place. (★☆☆)

```
# Author: Evgeni Burovski

Z = np.arange(11)
Z[(3 < Z) & (Z < 8)] *= -1
print(Z)
```

26. What is the output of the following script? (★☆☆)

```
# Author: Jake VanderPlas

print(sum(range(5),-1))
from numpy import *
print(sum(range(5),-1))
```


```
# Author: Jake VanderPlas

print(sum(range(5),-1))
from numpy import *
print(sum(range(5),-1))
```


27. Consider an integer vector Z, which of these expressions are legal? (★☆☆)

```
Z**Z
2 << Z >> 2
Z <- Z
1j*Z
Z/1/1
Z<Z>Z
```


```
Z**Z
2 << Z >> 2
Z <- Z
1j*Z
Z/1/1
Z<Z>Z
```


28. What are the result of the following expressions? (★☆☆)

```
np.array(0) / np.array(0)
np.array(0) // np.array(0)
np.array([np.nan]).astype(int).astype(float)
```

```
print(np.array(0) / np.array(0))
print(np.array(0) // np.array(0))
print(np.array([np.nan]).astype(int).astype(float))
```

29. How to round away from zero a float array ? (★☆☆)

```
# Author: Charles R Harris

Z = np.random.uniform(-10,+10,10)
print(np.copysign(np.ceil(np.abs(Z)), Z))

# More readable but less efficient
print(np.where(Z>0, np.ceil(Z), np.floor(Z)))
```

30. How to find common values between two arrays? (★☆☆)

```
Z1 = np.random.randint(0,10,10)
Z2 = np.random.randint(0,10,10)
print(np.intersect1d(Z1,Z2))
```

31. How to ignore all numpy warnings (not recommended)? (★☆☆)

```
# Suicide mode on
defaults = np.seterr(all="ignore")
Z = np.ones(1) / 0

# Back to sanity
_ = np.seterr(**defaults)

# Equivalently with a context manager
with np.errstate(all="ignore"):
    np.arange(3) / 0
```

32. Is the following expressions true? (★☆☆)

```
np.sqrt(-1) == np.emath.sqrt(-1)
```

```
np.sqrt(-1) == np.emath.sqrt(-1)
```

33. How to get the dates of yesterday, today and tomorrow? (★☆☆)

```
yesterday = np.datetime64('today') - np.timedelta64(1)
today     = np.datetime64('today')
tomorrow  = np.datetime64('today') + np.timedelta64(1)
```

34. How to get all the dates corresponding to the month of July 2016? (★★☆)

```
Z = np.arange('2016-07', '2016-08', dtype='datetime64[D]')
print(Z)
```

35. How to compute ((A+B)*(-A/2)) in place (without copy)? (★★☆)

```
A = np.ones(3)*1
B = np.ones(3)*2
np.add(A,B,out=B)
np.divide(A,2,out=A)
np.negative(A,out=A)
np.multiply(A,B,out=A)
```

36. Extract the integer part of a random array of positive numbers using 4 different methods (★★☆)

```
Z = np.random.uniform(0,10,10)

print(Z - Z%1)
print(Z // 1)
print(np.floor(Z))
print(Z.astype(int))
print(np.trunc(Z))
```

37. Create a 5x5 matrix with row values ranging from 0 to 4 (★★☆)

```
Z = np.zeros((5,5))
Z += np.arange(5)
print(Z)

# without broadcasting
Z = np.tile(np.arange(0, 5), (5,1))
print(Z)
```

38. Consider a generator function that generates 10 integers and use it to build an array (★☆☆)

```
def generate():
    for x in range(10):
        yield x
Z = np.fromiter(generate(),dtype=float,count=-1)
print(Z)
```

39. Create a vector of size 10 with values ranging from 0 to 1, both excluded (★★☆)

```
Z = np.linspace(0,1,11,endpoint=False)[1:]
print(Z)
```

40. Create a random vector of size 10 and sort it (★★☆)

```
Z = np.random.random(10)
Z.sort()
print(Z)
```

41. How to sum a small array faster than np.sum? (★★☆)

```
# Author: Evgeni Burovski

Z = np.arange(10)
np.add.reduce(Z)
```

42. Consider two random array A and B, check if they are equal (★★☆)

```
A = np.random.randint(0,2,5)
B = np.random.randint(0,2,5)

# Assuming identical shape of the arrays and a tolerance for the comparison of values
equal = np.allclose(A,B)
print(equal)

# Checking both the shape and the element values, no tolerance (values have to be exactly equal)
equal = np.array_equal(A,B)
print(equal)
```

43. Make an array immutable (read-only) (★★☆)

```
Z = np.zeros(10)
Z.flags.writeable = False
Z[0] = 1
```

44. Consider a random 10x2 matrix representing cartesian coordinates, convert them to polar coordinates (★★☆)

```
Z = np.random.random((10,2))
X,Y = Z[:,0], Z[:,1]
R = np.sqrt(X**2+Y**2)
T = np.arctan2(Y,X)
print(R)
print(T)
```

45. Create random vector of size 10 and replace the maximum value by 0 (★★☆)

```
Z = np.random.random(10)
Z[Z.argmax()] = 0
print(Z)
```

46. Create a structured array with `x` and `y` coordinates covering the [0,1]x[0,1] area (★★☆)

```
Z = np.zeros((5,5), [('x',float),('y',float)])
Z['x'], Z['y'] = np.meshgrid(np.linspace(0,1,5),
                             np.linspace(0,1,5))
print(Z)
```

47. Given two arrays, X and Y, construct the Cauchy matrix C (Cij =1/(xi - yj)) (★★☆)

```
# Author: Evgeni Burovski

X = np.arange(8)
Y = X + 0.5
C = 1.0 / np.subtract.outer(X, Y)
print(np.linalg.det(C))
```

48. Print the minimum and maximum representable value for each numpy scalar type (★★☆)

```
for dtype in [np.int8, np.int32, np.int64]:
   print(np.iinfo(dtype).min)
   print(np.iinfo(dtype).max)
for dtype in [np.float32, np.float64]:
   print(np.finfo(dtype).min)
   print(np.finfo(dtype).max)
   print(np.finfo(dtype).eps)
```

49. How to print all the values of an array? (★★☆)

```
np.set_printoptions(threshold=float("inf"))
Z = np.zeros((40,40))
print(Z)
```

50. How to find the closest value (to a given scalar) in a vector? (★★☆)

```
Z = np.arange(100)
v = np.random.uniform(0,100)
index = (np.abs(Z-v)).argmin()
print(Z[index])
```

51. Create a structured array representing a position (x,y) and a color (r,g,b) (★★☆)

```
Z = np.zeros(10, [ ('position', [ ('x', float, 1),
                                  ('y', float, 1)]),
                   ('color',    [ ('r', float, 1),
                                  ('g', float, 1),
                                  ('b', float, 1)])])
print(Z)
```

52. Consider a random vector with shape (100,2) representing coordinates, find point by point distances (★★☆)

```
Z = np.random.random((10,2))
X,Y = np.atleast_2d(Z[:,0], Z[:,1])
D = np.sqrt( (X-X.T)**2 + (Y-Y.T)**2)
print(D)

# Much faster with scipy
import scipy
# Thanks Gavin Heverly-Coulson (#issue 1)
import scipy.spatial

Z = np.random.random((10,2))
D = scipy.spatial.distance.cdist(Z,Z)
print(D)
```

53. How to convert a float (32 bits) array into an integer (32 bits) in place?

```
# Thanks Vikas (https://stackoverflow.com/a/10622758/5989906)
# & unutbu (https://stackoverflow.com/a/4396247/5989906)
Z = (np.random.rand(10)*100).astype(np.float32)
Y = Z.view(np.int32)
Y[:] = Z
print(Y)
```

54. How to read the following file? (★★☆)

```
1, 2, 3, 4, 5
6,  ,  , 7, 8
 ,  , 9,10,11
```

```
from io import StringIO

# Fake file
s = StringIO('''1, 2, 3, 4, 5

               6,  ,  , 7, 8

                ,  , 9,10,11
''')
Z = np.genfromtxt(s, delimiter=",", dtype=np.int)
print(Z)
```

55. What is the equivalent of enumerate for numpy arrays? (★★☆)

```
Z = np.arange(9).reshape(3,3)
for index, value in np.ndenumerate(Z):
    print(index, value)
for index in np.ndindex(Z.shape):
    print(index, Z[index])
```

56. Generate a generic 2D Gaussian-like array (★★☆)

```
X, Y = np.meshgrid(np.linspace(-1,1,10), np.linspace(-1,1,10))
D = np.sqrt(X*X+Y*Y)
sigma, mu = 1.0, 0.0
G = np.exp(-( (D-mu)**2 / ( 2.0 * sigma**2 ) ) )
print(G)
```

57. How to randomly place p elements in a 2D array? (★★☆)

```
# Author: Divakar

n = 10
p = 3
Z = np.zeros((n,n))
np.put(Z, np.random.choice(range(n*n), p, replace=False),1)
print(Z)
```

58. Subtract the mean of each row of a matrix (★★☆)

```
# Author: Warren Weckesser

X = np.random.rand(5, 10)

# Recent versions of numpy
Y = X - X.mean(axis=1, keepdims=True)

# Older versions of numpy
Y = X - X.mean(axis=1).reshape(-1, 1)

print(Y)
```

59. How to sort an array by the nth column? (★★☆)

```
# Author: Steve Tjoa

Z = np.random.randint(0,10,(3,3))
print(Z)
print(Z[Z[:,1].argsort()])
```

60. How to tell if a given 2D array has null columns? (★★☆)

```
# Author: Warren Weckesser

Z = np.random.randint(0,3,(3,10))
print((~Z.any(axis=0)).any())
```

61. Find the nearest value from a given value in an array (★★☆)

```
Z = np.random.uniform(0,1,10)
z = 0.5
m = Z.flat[np.abs(Z - z).argmin()]
print(m)
```

62. Considering two arrays with shape (1,3) and (3,1), how to compute their sum using an iterator? (★★☆)

```
A = np.arange(3).reshape(3,1)
B = np.arange(3).reshape(1,3)
it = np.nditer([A,B,None])
for x,y,z in it: z[...] = x + y
print(it.operands[2])
```

63. Create an array class that has a name attribute (★★☆)

```
class NamedArray(np.ndarray):
    def __new__(cls, array, name="no name"):
        obj = np.asarray(array).view(cls)
        obj.name = name
        return obj
    def __array_finalize__(self, obj):
        if obj is None: return
        self.info = getattr(obj, 'name', "no name")

Z = NamedArray(np.arange(10), "range_10")
print (Z.name)
```

64. Consider a given vector, how to add 1 to each element indexed by a second vector (be careful with repeated indices)? (★★★)

```
# Author: Brett Olsen

Z = np.ones(10)
I = np.random.randint(0,len(Z),20)
Z += np.bincount(I, minlength=len(Z))
print(Z)

# Another solution
# Author: Bartosz Telenczuk
np.add.at(Z, I, 1)
print(Z)
```

65. How to accumulate elements of a vector (X) to an array (F) based on an index list (I)? (★★★)

```
# Author: Alan G Isaac

X = [1,2,3,4,5,6]
I = [1,3,9,3,4,1]
F = np.bincount(I,X)
print(F)
```

66. Considering a (w,h,3) image of (dtype=ubyte), compute the number of unique colors (★★☆)

```
# Author: Fisher Wang

w, h = 256, 256
I = np.random.randint(0, 4, (h, w, 3)).astype(np.ubyte)
colors = np.unique(I.reshape(-1, 3), axis=0)
n = len(colors)
print(n)

# Faster version
# Author: Mark Setchell
# https://stackoverflow.com/a/59671950/2836621

w, h = 256, 256
I = np.random.randint(0,4,(h,w,3), dtype=np.uint8)

# View each pixel as a single 24-bit integer, rather than three 8-bit bytes
I24 = np.dot(I.astype(np.uint32),[1,256,65536])

# Count unique colours
n = len(np.unique(I24))
print(n)
```

67. Considering a four dimensions array, how to get sum over the last two axis at once? (★★★)

```
A = np.random.randint(0,10,(3,4,3,4))
# solution by passing a tuple of axes (introduced in numpy 1.7.0)
sum = A.sum(axis=(-2,-1))
print(sum)
# solution by flattening the last two dimensions into one
# (useful for functions that don't accept tuples for axis argument)
sum = A.reshape(A.shape[:-2] + (-1,)).sum(axis=-1)
print(sum)
```

68. Considering a one-dimensional vector D, how to compute means of subsets of D using a vector S of same size describing subset indices? (★★★)

```
# Author: Jaime Fernández del Río

D = np.random.uniform(0,1,100)
S = np.random.randint(0,10,100)
D_sums = np.bincount(S, weights=D)
D_counts = np.bincount(S)
D_means = D_sums / D_counts
print(D_means)

# Pandas solution as a reference due to more intuitive code
import pandas as pd
print(pd.Series(D).groupby(S).mean())
```

69. How to get the diagonal of a dot product? (★★★)

```
# Author: Mathieu Blondel

A = np.random.uniform(0,1,(5,5))
B = np.random.uniform(0,1,(5,5))

# Slow version
np.diag(np.dot(A, B))

# Fast version
np.sum(A * B.T, axis=1)

# Faster version
np.einsum("ij,ji->i", A, B)
```

70. Consider the vector [1, 2, 3, 4, 5], how to build a new vector with 3 consecutive zeros interleaved between each value? (★★★)

```
# Author: Warren Weckesser

Z = np.array([1,2,3,4,5])
nz = 3
Z0 = np.zeros(len(Z) + (len(Z)-1)*(nz))
Z0[::nz+1] = Z
print(Z0)
```

71. Consider an array of dimension (5,5,3), how to mulitply it by an array with dimensions (5,5)? (★★★)

```
A = np.ones((5,5,3))
B = 2*np.ones((5,5))
print(A * B[:,:,None])
```

72. How to swap two rows of an array? (★★★)

```
# Author: Eelco Hoogendoorn

A = np.arange(25).reshape(5,5)
A[[0,1]] = A[[1,0]]
print(A)
```

73. Consider a set of 10 triplets describing 10 triangles (with shared vertices), find the set of unique line segments composing all the triangles (★★★)

```
# Author: Nicolas P. Rougier

faces = np.random.randint(0,100,(10,3))
F = np.roll(faces.repeat(2,axis=1),-1,axis=1)
F = F.reshape(len(F)*3,2)
F = np.sort(F,axis=1)
G = F.view( dtype=[('p0',F.dtype),('p1',F.dtype)] )
G = np.unique(G)
print(G)
```

74. Given a sorted array C that corresponds to a bincount, how to produce an array A such that np.bincount(A) == C? (★★★)

```
# Author: Jaime Fernández del Río

C = np.bincount([1,1,2,3,4,4,6])
A = np.repeat(np.arange(len(C)), C)
print(A)
```

75. How to compute averages using a sliding window over an array? (★★★)

```
# Author: Jaime Fernández del Río

def moving_average(a, n=3) :
    ret = np.cumsum(a, dtype=float)
    ret[n:] = ret[n:] - ret[:-n]
    return ret[n - 1:] / n
Z = np.arange(20)
print(moving_average(Z, n=3))
```

76. Consider a one-dimensional array Z, build a two-dimensional array whose first row is (Z[0],Z[1],Z[2]) and each subsequent row is shifted by 1 (last row should be (Z[-3],Z[-2],Z[-1]) (★★★)

```
# Author: Joe Kington / Erik Rigtorp
from numpy.lib import stride_tricks

def rolling(a, window):
    shape = (a.size - window + 1, window)
    strides = (a.strides[0], a.strides[0])
    return stride_tricks.as_strided(a, shape=shape, strides=strides)
Z = rolling(np.arange(10), 3)
print(Z)
```

77. How to negate a boolean, or to change the sign of a float inplace? (★★★)

```
# Author: Nathaniel J. Smith

Z = np.random.randint(0,2,100)
np.logical_not(Z, out=Z)

Z = np.random.uniform(-1.0,1.0,100)
np.negative(Z, out=Z)
```

78. Consider 2 sets of points P0,P1 describing lines (2d) and a point p, how to compute distance from p to each line i (P0[i],P1[i])? (★★★)

```
def distance(P0, P1, p):
    T = P1 - P0
    L = (T**2).sum(axis=1)
    U = -((P0[:,0]-p[...,0])*T[:,0] + (P0[:,1]-p[...,1])*T[:,1]) / L
    U = U.reshape(len(U),1)
    D = P0 + U*T - p
    return np.sqrt((D**2).sum(axis=1))

P0 = np.random.uniform(-10,10,(10,2))
P1 = np.random.uniform(-10,10,(10,2))
p  = np.random.uniform(-10,10,( 1,2))
print(distance(P0, P1, p))
```

79. Consider 2 sets of points P0,P1 describing lines (2d) and a set of points P, how to compute distance from each point j (P[j]) to each line i (P0[i],P1[i])? (★★★)

```
# Author: Italmassov Kuanysh

# based on distance function from previous question
P0 = np.random.uniform(-10, 10, (10,2))
P1 = np.random.uniform(-10,10,(10,2))
p = np.random.uniform(-10, 10, (10,2))
print(np.array([distance(P0,P1,p_i) for p_i in p]))
```

80. Consider an arbitrary array, write a function that extract a subpart with a fixed shape and centered on a given element (pad with `fill` value when necessary) (★★★)

```
# Author: Nicolas Rougier

Z = np.random.randint(0,10,(10,10))
shape = (5,5)
fill  = 0
position = (1,1)

R = np.ones(shape, dtype=Z.dtype)*fill
P  = np.array(list(position)).astype(int)
Rs = np.array(list(R.shape)).astype(int)
Zs = np.array(list(Z.shape)).astype(int)

R_start = np.zeros((len(shape),)).astype(int)
R_stop  = np.array(list(shape)).astype(int)
Z_start = (P-Rs//2)
Z_stop  = (P+Rs//2)+Rs%2

R_start = (R_start - np.minimum(Z_start,0)).tolist()
Z_start = (np.maximum(Z_start,0)).tolist()
R_stop = np.maximum(R_start, (R_stop - np.maximum(Z_stop-Zs,0))).tolist()
Z_stop = (np.minimum(Z_stop,Zs)).tolist()

r = [slice(start,stop) for start,stop in zip(R_start,R_stop)]
z = [slice(start,stop) for start,stop in zip(Z_start,Z_stop)]
R[r] = Z[z]
print(Z)
print(R)
```

81. Consider an array Z = [1,2,3,4,5,6,7,8,9,10,11,12,13,14], how to generate an array R = [[1,2,3,4], [2,3,4,5], [3,4,5,6], ..., [11,12,13,14]]? (★★★)

```
# Author: Stefan van der Walt

Z = np.arange(1,15,dtype=np.uint32)
R = stride_tricks.as_strided(Z,(11,4),(4,4))
print(R)
```

82. Compute a matrix rank (★★★)

```
# Author: Stefan van der Walt

Z = np.random.uniform(0,1,(10,10))
U, S, V = np.linalg.svd(Z) # Singular Value Decomposition
rank = np.sum(S > 1e-10)
print(rank)
```

83. How to find the most frequent value in an array?

```
Z = np.random.randint(0,10,50)
print(np.bincount(Z).argmax())
```

84. Extract all the contiguous 3x3 blocks from a random 10x10 matrix (★★★)

```
# Author: Chris Barker

Z = np.random.randint(0,5,(10,10))
n = 3
i = 1 + (Z.shape[0]-3)
j = 1 + (Z.shape[1]-3)
C = stride_tricks.as_strided(Z, shape=(i, j, n, n), strides=Z.strides + Z.strides)
print(C)
```

85. Create a 2D array subclass such that Z[i,j] == Z[j,i] (★★★)

```
# Author: Eric O. Lebigot
# Note: only works for 2d array and value setting using indices

class Symetric(np.ndarray):
    def __setitem__(self, index, value):
        i,j = index
        super(Symetric, self).__setitem__((i,j), value)
        super(Symetric, self).__setitem__((j,i), value)

def symetric(Z):
    return np.asarray(Z + Z.T - np.diag(Z.diagonal())).view(Symetric)

S = symetric(np.random.randint(0,10,(5,5)))
S[2,3] = 42
print(S)
```

86. Consider a set of p matrices wich shape (n,n) and a set of p vectors with shape (n,1). How to compute the sum of of the p matrix products at once? (result has shape (n,1)) (★★★)

```
# Author: Stefan van der Walt

p, n = 10, 20
M = np.ones((p,n,n))
V = np.ones((p,n,1))
S = np.tensordot(M, V, axes=[[0, 2], [0, 1]])
print(S)

# It works, because:
# M is (p,n,n)
# V is (p,n,1)
# Thus, summing over the paired axes 0 and 0 (of M and V independently),
# and 2 and 1, to remain with a (n,1) vector.
```

87. Consider a 16x16 array, how to get the block-sum (block size is 4x4)? (★★★)

```
# Author: Robert Kern

Z = np.ones((16,16))
k = 4
S = np.add.reduceat(np.add.reduceat(Z, np.arange(0, Z.shape[0], k), axis=0),
                                       np.arange(0, Z.shape[1], k), axis=1)
print(S)

# alternative solution:
# Author: Sebastian Wallkötter (@FirefoxMetzger)

Z = np.ones((16,16))
k = 4

windows = np.lib.stride_tricks.sliding_window_view(Z, (k, k))
S = windows[::k, ::k, ...].sum(axis=(-2, -1))
```

88. How to implement the Game of Life using numpy arrays? (★★★)

```
# Author: Nicolas Rougier

def iterate(Z):
    # Count neighbours
    N = (Z[0:-2,0:-2] + Z[0:-2,1:-1] + Z[0:-2,2:] +
         Z[1:-1,0:-2]                + Z[1:-1,2:] +
         Z[2:  ,0:-2] + Z[2:  ,1:-1] + Z[2:  ,2:])

    # Apply rules
    birth = (N==3) & (Z[1:-1,1:-1]==0)
    survive = ((N==2) | (N==3)) & (Z[1:-1,1:-1]==1)
    Z[...] = 0
    Z[1:-1,1:-1][birth | survive] = 1
    return Z

Z = np.random.randint(0,2,(50,50))
for i in range(100): Z = iterate(Z)
print(Z)
```

89. How to get the n largest values of an array (★★★)

```
Z = np.arange(10000)
np.random.shuffle(Z)
n = 5

# Slow
print (Z[np.argsort(Z)[-n:]])

# Fast
print (Z[np.argpartition(-Z,n)[:n]])
```

90. Given an arbitrary number of vectors, build the cartesian product (every combinations of every item) (★★★)

```
# Author: Stefan Van der Walt

def cartesian(arrays):
    arrays = [np.asarray(a) for a in arrays]
    shape = (len(x) for x in arrays)

    ix = np.indices(shape, dtype=int)
    ix = ix.reshape(len(arrays), -1).T

    for n, arr in enumerate(arrays):
        ix[:, n] = arrays[n][ix[:, n]]

    return ix

print (cartesian(([1, 2, 3], [4, 5], [6, 7])))
```

91. How to create a record array from a regular array? (★★★)

```
Z = np.array([("Hello", 2.5, 3),
              ("World", 3.6, 2)])
R = np.core.records.fromarrays(Z.T,
                               names='col1, col2, col3',
                               formats = 'S8, f8, i8')
print(R)
```

92. Consider a large vector Z, compute Z to the power of 3 using 3 different methods (★★★)

```
# Author: Ryan G.

x = np.random.rand(int(5e7))

%timeit np.power(x,3)
%timeit x*x*x
%timeit np.einsum('i,i,i->i',x,x,x)
```

93. Consider two arrays A and B of shape (8,3) and (2,2). How to find rows of A that contain elements of each row of B regardless of the order of the elements in B? (★★★)

```
# Author: Gabe Schwartz

A = np.random.randint(0,5,(8,3))
B = np.random.randint(0,5,(2,2))

C = (A[..., np.newaxis, np.newaxis] == B)
rows = np.where(C.any((3,1)).all(1))[0]
print(rows)
```

94. Considering a 10x3 matrix, extract rows with unequal values (e.g. [2,2,3]) (★★★)

```
# Author: Robert Kern

Z = np.random.randint(0,5,(10,3))
print(Z)
# solution for arrays of all dtypes (including string arrays and record arrays)
E = np.all(Z[:,1:] == Z[:,:-1], axis=1)
U = Z[~E]
print(U)
# soluiton for numerical arrays only, will work for any number of columns in Z
U = Z[Z.max(axis=1) != Z.min(axis=1),:]
print(U)
```

95. Convert a vector of ints into a matrix binary representation (★★★)

```
# Author: Warren Weckesser

I = np.array([0, 1, 2, 3, 15, 16, 32, 64, 128])
B = ((I.reshape(-1,1) & (2**np.arange(8))) != 0).astype(int)
print(B[:,::-1])

# Author: Daniel T. McDonald

I = np.array([0, 1, 2, 3, 15, 16, 32, 64, 128], dtype=np.uint8)
print(np.unpackbits(I[:, np.newaxis], axis=1))
```

96. Given a two dimensional array, how to extract unique rows? (★★★)

```
# Author: Jaime Fernández del Río

Z = np.random.randint(0,2,(6,3))
T = np.ascontiguousarray(Z).view(np.dtype((np.void, Z.dtype.itemsize * Z.shape[1])))
_, idx = np.unique(T, return_index=True)
uZ = Z[idx]
print(uZ)

# Author: Andreas Kouzelis
# NumPy >= 1.13
uZ = np.unique(Z, axis=0)
print(uZ)
```

97. Considering 2 vectors A & B, write the einsum equivalent of inner, outer, sum, and mul function (★★★)

```
# Author: Alex Riley
# Make sure to read: http://ajcr.net/Basic-guide-to-einsum/

A = np.random.uniform(0,1,10)
B = np.random.uniform(0,1,10)

np.einsum('i->', A)       # np.sum(A)
np.einsum('i,i->i', A, B) # A * B
np.einsum('i,i', A, B)    # np.inner(A, B)
np.einsum('i,j->ij', A, B)    # np.outer(A, B)
```

98. Considering a path described by two vectors (X,Y), how to sample it using equidistant samples (★★★)?

```
# Author: Bas Swinckels

phi = np.arange(0, 10*np.pi, 0.1)
a = 1
x = a*phi*np.cos(phi)
y = a*phi*np.sin(phi)

dr = (np.diff(x)**2 + np.diff(y)**2)**.5 # segment lengths
r = np.zeros_like(x)
r[1:] = np.cumsum(dr)                # integrate path
r_int = np.linspace(0, r.max(), 200) # regular spaced path
x_int = np.interp(r_int, r, x)       # integrate path
y_int = np.interp(r_int, r, y)
```

99. Given an integer n and a 2D array X, select from X the rows which can be interpreted as draws from a multinomial distribution with n degrees, i.e., the rows which only contain integers and which sum to n. (★★★)

```
# Author: Evgeni Burovski

X = np.asarray([[1.0, 0.0, 3.0, 8.0],
                [2.0, 0.0, 1.0, 1.0],
                [1.5, 2.5, 1.0, 0.0]])
n = 4
M = np.logical_and.reduce(np.mod(X, 1) == 0, axis=-1)
M &= (X.sum(axis=-1) == n)
print(X[M])
```

100. Compute bootstrapped 95% confidence intervals for the mean of a 1D array X (i.e., resample the elements of an array with replacement N times, compute the mean of each sample, and then compute percentiles over the means). (★★★)

```
# Author: Jessica B. Hamrick

X = np.random.randn(100) # random 1D array
N = 1000 # number of bootstrap samples
idx = np.random.randint(0, X.size, (N, X.size))
means = X[idx].mean(axis=1)
confint = np.percentile(means, [2.5, 97.5])
print(confint)
```